UNITED STATES PATENT APPLICATION

of

Van Jacobson

Kathleen Nichols

and

Chandrashekhar Appanna

for

RED POLICERS

RED POLICERS

BACKGROUND OF THE INVENTION

Field of the Invention

The present invention pertains to data traffic management and in particular to data traffic policers.

Background Information

Packet-based data communication is a technique in which information at the source is formed into packets and transmitted through a medium, and at the destination, the packets are reassembled back to their original form. Each packet usually has a payload such as data, voice, image or any other type of information. Packet-based trafficking is very suitable for use in a network in that other than the particular time interval in which the network resources are dedicated to the transmission of the packet, the remaining time period is available for other packets to be transmitted. This time sharing arrangement along with the flexibility and speed of transmitting integrated information makes packet-based communication the standard method of communication used in the Internet.

Resources generally available to an Internet Service Provider (ISP) are limited. There is only a limited amount of bandwidth in which the ISP can channel packets through a network. Typically, an ISP used by a source station sends packets through the network to an ISP used by a destination station. A path through the worldwide Internet is typically supplied by a backbone provider. Within the network, a plurality of network devices (nodes) are connected together to forward the packets until eventually they arrive at their destination at an end station such as a client computer or a server. A node may be a router that processes and routes received packets to their appropriate destination. A router usually has multiple ingress/egress ports through which packets are channeled in

and out of the router. Because a router is only able to process a finite number of packets at a given time, when the traffic exceeds this bandwidth, congestion usually occurs. Often this congestion occurs at a border to an ISP, although congestion may also occur internal to the network. Congestion is usually alleviated by "dropping" packets. For instance, if a router receives more packets than it can handle, it simply "drops" the incoming packets until it regains the capacity to handle the packets. If the communication protocol used is the Transmission Control Protocol (TCP), the sending node is usually notified when a packet is received (ACK message) and times out when it does not receive an ACK because a packet was dropped. Timing out usually causes the sending node to transmit the packets at a lower transmission rate. Because the transmission is duplicated for the lost packets, there occurs a higher consumption of the already limited bandwidth and a further deterioration in the efficacy of the network. Furthermore, the retransmitted packets are delivered late, if at all, slowing down the interchange of information.

From the ISP's point of view, it is desirable to manage the network so as to provide a better and a more predictable service in terms of bandwidth, latency and loss characteristics. In many instances and for various reasons, a user and the ISP will enter into a "traffic contract" that sets forth a rate in which the user may transmit data to the ISP's network. The term "traffic contract" is used to mean any agreement or decision that traffic through a particular network device is to be limited to a rate less than the wire rate, where the wire rate is the maximum rate which the physical interconnection can deliver. For example, the traffic contract may be between a customer and an ISP to limit the rate which the customer delivers packets (or bytes) to the ISP, or the rate at which the ISP delivers packets to the customer. Further, the traffic contract may be used in an enterprise network in order to limit certain types of traffic into or out of the network, for example between hosts and "the network" in order to meet traffic engineering goals. In the enterprise network example, there is no "contract" between an end user and someone else, the contract is simply how the network is set up. As a further example, a traffic contract may be involved in setting up a quality of service (QoS) arrangement with an end user. As a further example, the traffic contract rate may be to set a limit on the amount of traffic which gets special treatment in the network, or just to limit the amount of traffic in general. Also, the ISP may have a traffic contract with the provider of the network back-

bone. It is usually the burden of the user to shape the traffic to meet the contracted rate. However, the ISP, the backbone (or forwarder) provider, the customer, the parts of an enterprise network, etc. may place a "policer" at ports of his exit nodes to enforce the contract.

The policer monitors its incoming packets to determine if they conform to the contract. If a packet is non-conforming (i.e., exceeds the contract rate), the policer may simply "drop" the packet. In many non-ideal implementations, a policer may be implemented to drop packets in a manner that is similar to that of a "tail-drop" in a traffic congestion situations. Tail-dropping occurs when a node is not able to handle any more incoming packets. For example, a queue simply fills up from a burst and drops all later arriving packets. A disadvantage pertaining to tail dropping is that it is very unfair. Stated differently, different connections may not have their packets dropped proportionally according to their usage. It is desirable to have a traffic policing method which does not do "tail dropping".

On a different note, one known traffic-policing algorithm is the "leaky bucket" algorithm (for example as used in the Asynchronous Transfer Mode (ATM) Protocol, and as described in the ATM Forum's Traffic Management Specification Version 4.1. ATMs forward fixed size packets known as "cells." A continuous-state leaky bucket algorithm, as its name implies, can be imagined as a finite-capacity bucket (actually a queue or a counter) in which a real-valued content drains out at a continuous rate of 1 unit of content per time-unit and whose content is increased by the increment I for each conforming cell. The leaky bucket algorithm is fully described by Andrew S. Tanenbaum in his book *Computer Networks, Third Edition*, published by Prentice Hall, Copyright date 1996, all disclosures of which are incorporated herein by referenced, particularly at pages 380-381. As shown in Fig. 1, at block 100, the algorithm is activated when a cell is received. At its initiating state, the content of the bucket is zero. With the arrival of the first cell ta(1), the Last Conformance Time (LCT) is set to ta(1). With the successive arrival of the cells such as the kth cell at time ta(k), at block 102, the content of the bucket x' is updated to equal to the value of the leaky bucket at the arrival of the last conforming cell minus the amount the bucket has drained since that arrival. Note that the content of the bucket cannot be less than zero and at blocks 104-106, if the content of the bucket x' is less than

zero, the value x' is adjusted to zero. At block 108, if the value x' is greater than a limit value L, the cell is non-conforming and at block 110, the values of x' and LCT remain unchanged. Otherwise if the value x' is less than or equal to the limit value L, the cell is conforming and at block 112, the bucket content x is set to x' plus the increment I for that current cell and the LCT is set to ta(k). Further details may be found in the forum paper specified above.

The leaky bucket algorithm described above, however, tail drops "bursts" that may occur in traffic. The bucket fills, and before it empties more packets arrive because of the burst. After the bucket fills, all subsequent packets are simply discarded since they arrive faster than the bucket empties.

In another known example, a dual-leaky bucket is used to accommodate for the data bursts. The first leaky bucket polices the cells for conformance to the sustained cell rate as agreed in the contract. The second leaky bucket polices the cells for compliance with the maximum burst size allowable by the contract. Besides the leaky bucket approach, other approaches may be used such as the virtual scheduling algorithm using theoretical arrival time (TAT) also described in the forum paper. However, the virtual scheduling algorithm, too, suffers the drawbacks of tail dropping in bursty traffic.

There is needed a traffic policer which drops packets from flows in proportion to the amount of bandwidth used by the flows so that the dropping is fair, and also which does not tail drop when receiving bursty flows.

SUMMARY OF THE INVENTION

A Random Early Detection (RED) policer in accordance with the invention permits bursty traffic and does not tail-drop arriving packets. The policer uses randomization in choosing which packets to drop. With randomization the probability of dropping a packet from a particular sending node is roughly proportional to the node's bandwidth share, hence the invention is fair to nodes using different amounts of bandwidth.

According to one embodiment, the RED policer can be viewed as controlling a virtual queue in which its capacity limit is determined by a virtual time debt. A virtual time debt for each packet is a difference between the real time of a packet arrival and the theoretical (virtual) time the packet should have arrived. The time that the packet should have arrived is given by the traffic contract. The RED policer calculates a filtered virtual time debt, for example by using an Exponential Weighted Moving Average (EWMA) filter. When the filtered virtual time debt exceeds some predetermined minimum threshold, the RED policer drops the next packet and then starts to randomly drop packets based on a probability determination. That is, the drop probability increases with increasing filtered virtual time debt, and all packets are dropped once the filtered virtual time debt reaches an upper threshold.

BRIEF DESCRIPTION OF THE DRAWINGS

The invention description below refers to the accompanying drawings, of which:

Fig. 1 is a flowchart of a conventional continuous-state leaky bucket algorithm;

Fig. 2 is an exemplary network having a plurality of nodes in which the present invention may be implemented;

Fig. 3 is a schematic diagram of a router including a policer constructed in accordance with the invention;

Fig. 4 is a schematic diagram of a Random Early Detection (RED) policer implemented in a router;

Fig. 5 is a flow diagram of an exemplary RED policer;

Fig. 6 is a graph showing a relationship between the filtered time debt and the packet drop probability; and

Fig. 7 is a schematic diagram of a router having a plurality of policers constructed in accordance with the invention and wherein, packets are channeled to the policers by a packet classifier.

DETAILED DESCRIPTION OF AN ILLUSTRATIVE EMBODIMENT

An exemplary network 200 as shown in Fig. 2 illustrates a plurality of nodes 220, 300 such as routers that forward packets in the network to their various destinations. In one instance, nodes 300 at the outer edges of the network have policers that monitor the entering traffic. These policers detect violations in actual traffic flow as compared with a setup rate and penalize accordingly. In another instance, nodes 220 have policers that monitor the traffic entering the nodes. The policer in the node may regulate the traffic flow to ensure that the node does not get overflowed.

Fig. 3 illustrates a node such as a router 300 with a policer 400 that is constructed in accordance with the invention. For purposes of illustration, only one policer is shown although the router may have a plurality of policers that may police the traffic according to the entering packet classification, for example. The router 300 in its general form comprises a processing unit 312 and a memory unit 314 coupled together by a bus or a switch fabric 316. Further coupled to the switch fabric may be a plurality of input/output (I/O) interfaces 318 that interact with other nodes in the network. In one example, an operating system (OS) 320 resides in the memory unit 314 along with the policer 400. Together, they facilitate in policing the entering packets when executed by the processing unit 312.

The memory unit 314 may be a volatile memory such as a Dynamic Random Access Memory (DRAM). The policer 400 may also reside in a non-volatile memory such as a Read Only Memory (ROM) or a Flash memory. Further, the policer may be stored in a storage medium such as magnetic or optical disks. Collectively, the mentioned memories, storage mediums and the like will be referred to as a processor executable memory. Additionally, the policer may be implemented in hardware such as an application specific integrated circuit (ASIC).

As shown in Fig. 4 in more detail, the policer 400 based on Random Early Detection (RED) uses a running estimate (which in one instance, is based on an exponential weighted moving average (EWMA)) of the average packet flow and is shown as filter block 404. Because the drop policy is based on the filtered packet rate, this feature allows the policer to absorb traffic bursts without dropping the last packets of the burst, commonly called the "tail dropping" problem. Nodes within a network inherently generate bursty traffic. Policers using token buckets or virtual scheduling algorithms usually drop the last arriving packets, that is they tail drop the bursts.

An advantage of the RED policer 400 is that it is fair when dropping packets because that decision is based on randomness. The RED policer 400 uses randomization in choosing which arriving packets to drop; with this method, the probability of dropping a packet from a particular sending node is roughly proportional to that node's share of the bandwidth. Fairness arises from the fact that the randomness "samples" the input stream, and thus if a particular stream's packets appear in the stream more frequently, they will be "sampled" more frequently and therefore dropped more frequently.

One option is to measure in bytes rather than in packets. Measurement in bytes allows more accurate measurement of the actual filtered virtual time debt, and thus allows a more accurate response to increasing filtered virtual time debt.

Briefly, when the filtered packet flow rate passes a lower threshold, the RED policer drops arriving packets randomly with a low probability. The drop probability increases with increasing filtered packet flow rate and all packets are dropped once the filtered packet flow rate reaches an upper threshold. However, the upper threshold is not usually reached because the RED's regulating characteristic matches the input rate with

the output rate based on a control law similar to that found in a closed-loop servo system. The control law block is shown as block 406. Though not necessary, the RED policer 400 is suitable in a network where the transmission protocol responds to the dropped packets as indications that the transmission rate should be lowered. That is, a packet dropped by a RED policer causes a source station using an adaptive flow technique such as TCP/IP to reduce its transmission rate.

Another feature is that the policer using RED need not be tightly coupled to packet forwarding and its computations do not have to be made in the time-critical packet forwarding path. Much of the work such as the computation of the filtered packet flow rate and of the packet-dropping probability may be performed in parallel with the packet forwarding, or may be computed as a low-priority task. Thus, RED can be adapted to increasingly-high-speed output lines.

The use of a sampler 402 leads to a simpler forwarding path, better parameter settings of the filter block 404 and an architecture that lends itself to high-speed implementations. The sampling and the filtering may be performed at intervals that are either fixed or random. At each sample, the control law block 406 uses the filtered value from the filter block 404 to decide whether and when to drop an arriving packet. As will later become apparent, the sampling time is a factor that determines the gain of the filter. When the control law block 406 determines that a packet is to be dropped, the policer 400 sets a counter 408, whose operation will be further described below.

The RED policer 400 can be viewed as controlling a virtual queue in which its capacity limit is determined by a virtual time debt. A virtual time debt is a difference between the real time of a packet arrival and the theoretical (virtual) time the packet should have arrived (for instance, the virtual time of packet arrival may be the contracted packet rate between a user and an ISP, or between an ISP and a backbone provider).

The RED policer 400's filter block 404 provides the filtering operation that calculates the virtual time debt at intervals of sample time T. As an example, the filter operation may be based on an EWMA low-pass filter, which is expressed as:

$$F_k = (1 - g)F_{k-1} + g(V_T\text{-}now) \qquad \text{EQUATION 1}$$

where g is the gain of the filter and having a value $0 < g < 1$, F_k is the filtered virtual time debt at sample time k, F_{k-1} is the filtered virtual time debt at sample time k-1 and $V_{T\text{-}now}$ is the virtual time debt at sample k.

In particular, the filter uses a gain value g, wherein g is the inverse of the sample time in a round-trip Internet time. For instance, one round-trip time is where when a router drops a packet, the TCP receiver fails to receive a packet, does not send an ACK, and the sender times out. The sender then retransmits the missing packet to the TCP receiver. The rationale is that the filter will average over a round-trip time and approximate the mean over the round-trip time. If the average sample interval is the transmission time of an MTU, the gain should be the inverse of the bandwidth of the MTU sized packets. Because the actual-round trip time of any connection is difficult to obtain, a canonical value such as a 100 milliseconds may be used. For computational efficiency, the gain is rounded to the nearest power of two.

The gain should typically be set in a manner suitable for the particular connection. Note that if the gain is set too small in comparison with the inverse of the bandwidth, the EWMA filter would be too slow to respond to accumulation of the virtual time debt. On the other hand, if the gain is too large, it causes the EWMA filter to respond too quickly resulting in unnecessary packets being dropped. For additional reading concerning the sampling time interval and the gain of the EWMA filter, see V. Jacobson , K. Nichols, K. Poduri, "*RED in a Different Light*", not published but widely circulated.

For example, a value of g of 0.01 has been found suitable for many applications, a value of 0.01 for g permits the new information to affect the accumulated value by only 1%.

In summing, the RED policer may be constructed in the following manner. Its first component calculates the single packet virtual time debt by the formula:

$$\text{Time debt} = \text{expected packet arrival time} - \text{actual packet arrival time}$$

EQUATION 2

The time debt is represented by F_R for the k^{th} received packet, and the EWMA is computed using Equation 1.

Its other component calculates the packet-dropping probability, which in turn determines whether and when to drop packets, given the traffic flow. So far, the first component of the RED policer has been discussed. Concerning the latter component of random dropping, it is desired to drop the packets at random intervals and from randomly chosen flows in order to avoid unfairness, and to drop packets with sufficiency as to regulate the traffic flow.

The RED policer randomly drops packets when the virtual time debt exceeds some predetermined minimum threshold for example, as given by Equation 1. As shown in Fig. 5, in block 502, the RED policer calculates the filtered virtual time debt. In block 504, the filtered virtual time debt is compared with a minimum threshold. If the filtered virtual time debt is less than the minimum threshold, in block 506, no packets are dropped. In block 506, if the filtered virtual time debt exceeds a certain maximum threshold, in block 508, all packets are dropped until the filtered time debt falls below the maximum threshold. Else if the filtered virtual time debt is between the minimum threshold and the maximum threshold, in block 510, the RED policer generates a pseudo-random number based on the level the filtered virtue time debt exceeds the minimum threshold and sets a counter (such as the one shown in Fig. 4) with the pseudo-random number in block 512. Note that in blocks 512-514, the counter once set begins to count down with each arriving packet and once the counter reaches zero, the corresponding packet is dropped and the counter is then reset based on the activities in blocks 502-510. Usually the maximum threshold will be not reached if all the sending nodes are cooperative in reducing the packet transmission as the packets are dropped between the minimum threshold and the maximum threshold; this ensures that the filtered virtual time debt does not significantly exceed the minimum threshold.

As mentioned above, random packet-dropping is based on the level the filtered virtual time debt has exceeded the minimum threshold. As an example, referring to the graph on Fig. 6, the RED policer may determine that the probability of dropping a packet is 30 percent based on the level the filtered virtual time debt has exceeded the minimum

threshold. Then, the inverse of 30 percent rounded to the nearest number is the range the random number will be selected to set the counter. In this instance, the random number will be from one to three. Suppose that the random number generated is two, then the RED policer sets the counter to two, which is decremented with each incoming packet. When the counter reaches zero, the corresponding incoming packet is dropped. Note the relationship that the higher the filtered virtual time debt exceeds the minimum threshold the higher the packet drop probability and hence the increase in packet drop frequency. Note that this method could be used to control the virtual time debt size even if the sending node fails to reduce its throughput in response to the dropped packets.

Fig. 7 is a schematic diagram of a plurality of policers used to police an arbitrary mix of data traffic flowing in a wire. The data packets are passed through a packet classifier 702 that determines which packet should go to which policer 400. For instance, the packet classification may be based on the source address of the packet. Each policer may have a pre-programmed virtual time debt that corresponds to the contracted rate of the source of the packets. Once the packets enter the policers 400, the operation of each policer will be similar to that described with respect to Fig. 4

An improved policer based on RED has been described. It will however be apparent that other variations and modifications may be made to the described embodiments, with the attainment of some or all of their advantages. Therefore, it is the object of the appended claims to cover all such variations and modifications that come within the true spirit and scope of the invention.

What is claimed is: